ДОДАТОК А

Вихідний код програми

```
import re
import pandas as pd
import sklearn
from spacy.matcher import Matcher
import networkx as nx
import matplotlib.pyplot as plt
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
import re
import string

from google.colab import drive
drive.mount('/content/drive')
import pandas as pd
data = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/1.txt"
)
data.head(10)

import spacy
nlp = spacy.load("en_core_web_sm")
doc = nlp(data)
print("TEXT\t LEMMA\t POS\t TAG\t DEP")
for token in doc:
    print(token.text +"\t", token.lemma_ +"\t", token.pos_ +"\t",
 token.tag_ +"\t", token.dep_ +"\t")

tokenizer = nltk.RegexpTokenizer(r"\w+")
new_words = tokenizer.tokenize(text)
token_list = []
for token in new_words:
    token_list.append(token)
filtered_sentence =[]
for word in token_list:
    lexeme = nlp.vocab[word]
    if lexeme.is_stop == False:
        filtered_sentence.append(word)
print("Source")
print(token_list)
print("Without stop words")
print(filtered_sentence)

from nltk.stem.porter import *
```

```
stemmer = PorterStemmer()
for token in filtered_sentence:
    print(token + ' --> ' + stemmer.stem(token))
def get_frequency_matrix(sentences):
    frequency_matrix = {}
    ps = PorterStemmer()
    stopWords = set(stopwords.words("english"))
    for sent in sentences:
        table = {}
        words = nltk.word_tokenize(sent)
        for word in words:
            word = word.lower()
            word = ps.stem(word)
            if word in stopWords:
                continue
            if word in table:
                table[word] += 1
            else:
                table[word] = 1
        frequency_matrix[sent[:100]] = table
    return frequency_matrix
print(get_frequency_matrix(new_words))
print()
print(get_frequency_matrix(filtered_sentence))

from spacy import displacy
sentence_spans = list(doc.sents)
svg = displacy.render(sentence_spans, style='ent', jupyter=True,
options={'distance': 90,'compact':True})


def extract_entity(sent):
  entity1 = ""
  entity2 = ""
  prev_token_t = ""
  prev_token_d = ""
  prefix = ""
  modifier = ""
  for tokens in nlp(sent):
    if tokens.dep_ != "punct":
      if tokens.dep_ == "compound":
        prefix = tokens.text
        if prev_token_d == "compound":
          prefix = prev_token_t + " "+ prefix
      if tokens.dep_.endswith("mod")==True:
        modifier = tokens.text

        if prev_token_d == "compound":
          modifier = prev_token_t + " " + modifier
```

```
      if tokens.dep_.find("subj") == True:
        entity1 =modifier+ " " + prefix + " " + tokens.text
        prefix = ""
        modifer = ""
          if tokens.dep_.find("obj") == True:
        entity2 =modifier+ " " + prefix + " " + tokens.text
        prefix = ""
        modifer = ""
      prev_token_t = tokens.text
      prev_token_d = tokens.dep_

  return [entity1.strip(), entity2.strip()]

entity_pairs = []
for i in tqdm(data['sentence']):
  entity_pairs.append(extract_entity(i))
tqdm._instances.clear()
subjects = []
objects = []
subjects = [x[0] for x in entity_pairs]
objects = [x[1] for x in entity_pairs]
relations = [extract_relation(i) for i in data['sentence']]
relations

import nltk
nltk.download('punkt')
nltk.download('stopwords')
def NotStopWord(word):
    return word not in stopwords.words('english')

def preprocess(sent):
  sent = re.sub("[\(\[].*?[\)\]]", "", sent)
  tokens = []
  temp = ""
  words = word_tokenize(sent)
  punctuations = '"#$%&\'()*+,-/:;<=>@\\^_`{|}~'
  words = map(lambda x: x.translate(str.maketrans('','',punctuati
ons)), words)
  words = map(str.lower,words)
  words = filter(lambda x: NotStopWord(x),words)
  tokens = tokens + list(words)
  temp = ' '.join(word for word in tokens)
  return temp
preprocessed_data = [preprocess(i) for i in (data['sentence'])]
```

ДОДАТОК Б

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	93 с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата			
Розробив		Громак О.В.			Дослідження та застосування графів знань та нейромережевих моделей для обробки природномовних текстів	Шифр групи	Код напр./спец.
Перевірив		Рябова Н. В.				СШІм-20-3	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Філатов В.О.					